Why Multi-Step Tool-Use Reinforcement Learning Collapses and How Supervisory Signals Fix It

Yupu Hao [1 2] **Zhuoran Jin** [1 2] **Huanxuan Liao** [1 2] **Kang Liu** [1 2] **Jun Zhao** [1 2]

Abstract

Tool use enables large language models (LLMs) to perform complex tasks, and recent agentic reinforcement learning (RL) methods show promise for enhancing model capabilities. However, RL alone often leads to instability or limited gains in tool-use tasks. In our experiments, some models exhibit catastrophic collapse, where performance abruptly drops and tool-invocation structures fail. The analysis reveals that these failures stem from unexpected probability spikes in specific control tokens, disrupting structured execution, yet the underlying tool-use capability remains intact, merely obscured by specific formats. To address this, we systematically investigate a diverse set of supervisory signals, including off-policy supervision, hint-based guidance, erroneous example supervision, and others, applied under both synchronous and interleaved training schemes. We find that interleaving supervised fine-tuning (SFT) with RL substantially improves stability, but exhibits degraded performance under Format and Content out-of-distribution (OOD) evaluation. We also analyze the impact of learning rates and generalization across settings. These results highlight the importance of understanding RL failures and demonstrate how diverse supervisory signals can guide exploratory learning, enabling robust training of LLMs for complex, multi-step tool-use tasks. Our Code is available at `https://github.com/hypasd-art/Tool-RL-Box`.

[1]Institute of Automation, Chinese Academy of Sciences, Beijing, China [2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China. Correspondence to: Jun Zhao <jzhao@nlpr.ia.ac.cn>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

1. Introduction

Tool use plays an increasingly important role in large language models (LLMs) (Li et al., 2023; Mialon et al., 2023; Plaat et al., 2025). By leveraging external tools, LLMs can function as intelligent agents capable of autonomously performing complex interactive tasks (Qu et al., 2025; Yao et al., 2024). However, the multi-step and structured nature of tool-based interactions, together with the diversity of tool feedback, introduces substantial challenges for improving model capabilities. To mitigate these issues, prior work has focused on improving tool-use performance through optimized interaction frameworks (Qin et al., 2024), large-scale synthesis of high-quality trajectories (Liu et al., 2025b; Prabhakar et al., 2025b), and refined supervised or unsupervised training methodologies (Liu et al., 2025b; Lu et al., 2025). Particularly recent advances in agentic RL (Lu et al., 2025; Wang et al., 2025; Mai et al., 2025) have motivated growing interest in RL as a more principled framework for agentic interaction, as agentic RL has demonstrated strong performance across a variety of complex tasks (Zhang et al., 2025b; Wu et al., 2025).

In our experiments, we identify two major challenges in multi-turn agentic RL: **training instability** and **limited performance gains**. Instabilities often manifest as catastrophic collapse, where model performance abruptly drops to zero at specific training steps, frequently accompanied by sharp spikes in KL divergence. Limited performance gains occur when the model avoids collapse but remains on prolonged performance plateaus, exhibiting minimal improvement despite continued training. Both failure modes hinder learning and compromise the model's ability to generate valid tool-use behaviors. To investigate the underlying causes, we analyze the training dynamics at the token level. By tracking model-generated trajectories throughout training, we observe that the frequency of specific tokens, typically special control tokens such as tool-call start or end markers or finish markers, can spike unexpectedly, disrupting the intended tool-invocation structure or agent response and leading to invalid or degenerate interaction patterns. As illustrated in Figure 1, the frequency of erroneous responses can surge sharply at certain training stages. However, **these phenomenon do not indicate a loss of the model's un-**

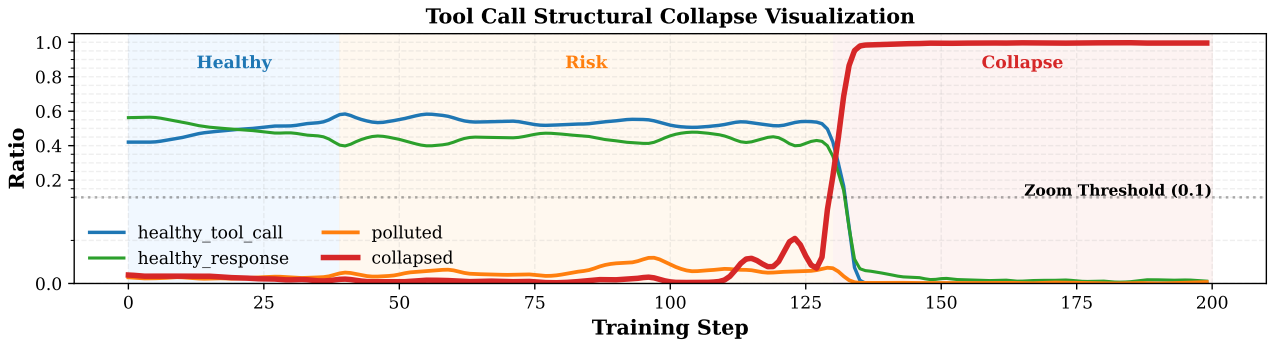


Figure 1 The changes of token appearing frequency in the RL training process of model Qwen2.5-1.5B-Instruct. Polluted indicates that irrelevant special tokens appear in the output, whereas Collapsed indicates that the output is completely invalid or incorrect.

derlying capability, but only affects performance on the specific tool-use prompt at hand.

These observations indicate that RL alone is insufficient for stable, long-horizon structured generation. We find supervised fine-tuning (SFT) on high-quality tool-use trajectories provides a strong initialization, improving early performance and preventing collapse. Nonetheless, prior work suggests that models trained with SFT can be limited in terms of generalization and performance compared to RL (Chu et al., 2025). Therefore, we aim to **explore the effects of different post-training methods and attempt to combine them to mitigate their shortcomings**. Yet, a key open question remains: *how should supervisory signals be integrated into RL training, and which forms of supervision are most effective for enhancing stability?*

To address this question, we conduct a systematic investigation of diverse supervisory interventions, such as off-policy supervision, hint-based guidance, and learning from erroneous examples. These interventions can be categorized into two paradigms: **synchronous training**, where supervisory signals are incorporated directly into the RL process within a single training stage, and **interleaved training**, where information from the RL process is first processed to generate supervisory signals, which are then used for instruction fine-tuning and interleaved with subsequent RL updates.

Our results show that interleaved training is more stable than synchronous training, which can even underperform direct RL due to low-probability supervisory tokens that misalign with the model's online sampling distribution. Furthermore, on Qwen2.5-1.5B-Instruct, our proposed *reflection signal*, extracted from intermediate reasoning steps, further improves both stability and final performance. These findings suggest that many useful signals are difficult to learn from sparse rewards alone, while explicitly extracting and fine-tuning them can significantly accelerate tool-learning.

We also observe several key factors: (1) **Higher learning rate in the SFT stage**: A larger learning rate is needed in SFT to ensure stability compared to RL. Low learning rates in either SFT or interleaved training lead to poor performance, whereas overly high rates can cause overfitting. (2) **RL performance collapse is format-specific rather than indicative of a general loss of capability**: Performance collapse during RL (direct RL, off-policy supervision, or hint-based guidance) occurs mainly on the same training distribution of format. Even models that collapse on the test set show minimal degradation in OOD settings, where unseen tools are invoked in novel formats. In contrast, models that perform well in-distribution but include a separate SFT stage often show significant OOD performance drops. Together, these results highlight the importance of understanding and leveraging supervisory signals to enable stable, high-performing agentic RL, providing a framework for training LLMs on complex, multi-step tool-use tasks.

Our work makes three key contributions:

- **We identify structural collapse as a fundamental failure mode in multi-turn agentic RL.** We show that RL on tool-use tasks does not primarily degrade reasoning ability, but induces a *structural collapse* in which generation degenerates into malformed control-token sequences, breaking tool-invocation structure while preserving underlying task competence.
- **We uncover a token-level mechanism underlying RL instability.** We find that RL disproportionately amplifies special control tokens, leading to a redistribution of policy mass. This reveals that instability arises from *control-token dynamics* rather than capability degradation, providing a mechanistic explanation for collapse.
- **We conduct a systematic study of supervisory signals for stabilizing multi-turn agentic RL.** We systematically explore six supervisory configurations across two model families and categorize them into synchronous and interleaved paradigms. We further introduce Process Reflection Supervision, which converts intermediate trajectory information into textual

supervisory signals. Empirically, interleaved training consistently improves stability and generalization, while synchronous methods are prone to distribution mismatch.

2. Related Works

2.1. Tool Learning

Recent advances have extended LLMs with tool-use capabilities (Hao et al., 2025; Prabhakar et al., 2025a; Qin et al., 2024), enabling interaction with external APIs and environments beyond text generation (Lu et al., 2025). Through tool invocation, models can perform task execution (Li et al., 2023; Trivedi et al., 2024), information retrieval (Jin et al., 2025), and planning (Xie et al., 2024), improving their ability to solve complex tasks efficiently. Prior work has enhanced tool-use capabilities through reasoning frameworks (Qin et al., 2024; Yao et al., 2023), supervised fine-tuning (Lu et al., 2025; Zhang et al., 2025d; Li et al., 2025), and reinforcement learning (RL) (Srivastava & Aggarwal, 2025; Hu, 2025; Shao et al., 2024). Although RL improves tool-invocation performance (Qian et al., 2025; Jiang et al., 2025; Xue et al., 2025), its effectiveness strongly depends on the base model's prior tool knowledge. Smaller or weakly initialized models often fail to benefit from RL, while high-quality SFT data alone can already provide strong performance. This motivates incorporating supervisory signals into RL to stabilize multi-turn interactions and improve tool-use learning.

2.2. Expert Trajectories in Reinforcement Learning

RL improves models through trajectory exploration and reward-based updates (Shao et al., 2024; Srivastava & Aggarwal, 2025), but training can stagnate when sampling quality is poor. Recent work addresses this issue by integrating expert or ground-truth trajectories into RL (Fu et al., 2025; Liu et al., 2025a). For example, LUFFY (Yan et al., 2025) partially replaces sampled trajectories with expert ones and reweights low-probability tokens, while ReLIFT (Ma et al., 2025) alternates RL with targeted SFT. Other methods guide generation using partial correct answers (Zhang et al., 2025a;c; Trivedi et al., 2024; Huang et al., 2025; Zhan et al., 2025) or retrieved memories and experiences (Chen et al., 2025; Fang et al., 2025). However, these approaches are mainly studied in single-turn reasoning tasks such as mathematics. Their effectiveness for multi-turn tool-use scenarios remains underexplored.

3. Preliminary

We represent multi-turn tool utilization trajectory τ by LLMs as a sequence of actions and feedbacks: $\tau = (a_1, r_1, a_2, r_2, \ldots, a_T, r_T)$. Each action $a_t \in \mathcal{A}$ is drawn from an action space $\mathcal{A}$ that includes both **feasible tool invocations** and **natural language responses**. At each step, the feedback $r_t \in \mathcal{R}$ is generated as a function of the current action and the preceding trajectory:

$$r_t = R(a_t \mid a_1, r_1, \ldots, a_{t-1}, r_{t-1}) \in \mathcal{R} \tag{1}$$

where $\mathcal{R}$ denotes the feedback space, encompassing both **environment responses** (e.g., tool execution outputs) and **user responses**.

Conditioned on the accumulated interaction history, the model samples the next action according to:

$$a_t \sim P(a_t \mid a_1, r_1, \ldots, a_{t-1}, r_{t-1}) \tag{2}$$

4. Method

In this section, we first analyze the *training instability and limited performance gains* observed under pure reinforcement learning settings. By examining model-generated trajectories during training, we identify the underlying causes of these failures. In contrast, introducing supervised fine-tuning prior to RL yields substantially improved performance, underscoring the critical role of supervisory signals. To further understand how such signals can be effectively incorporated into the training process, we investigate the impact of different types of supervision and their integration strategies on RL training.

4.1. Experiment Setup

Dataset. We conduct experiments on BFCL-V3 (Patil et al., 2025), a multi-turn tool-use benchmark where models must invoke multiple tools across interactive environments to answer user queries and follow-up questions. We focus on four challenging settings: *Base*, *Miss Func*, *Miss Param*, and *Long Context*. For training, we randomly sample 100 instances from each of the first three settings, resulting in 300 trajectories, while reserving the remaining data for evaluation. Due to excessive context length, *Long Context* is excluded from training. We also augment supervised fine-tuning with converted ToolACE (Liu et al., 2025b) data to study the impact of training distribution.

Model. For our analysis experiments, we use Qwen2.5-1.5B-Instruct (Team et al., 2024) and Qwen3-1.7B (Yang et al., 2025) as the base model. During the SFT stage, we decompose each multi-turn interaction into multiple single-turn instances and train the model on these isolated steps. In contrast, during RL, we adopt a full multi-turn tool-use process, where the model is required to complete the entire interaction trajectory, including all tool invocations, to

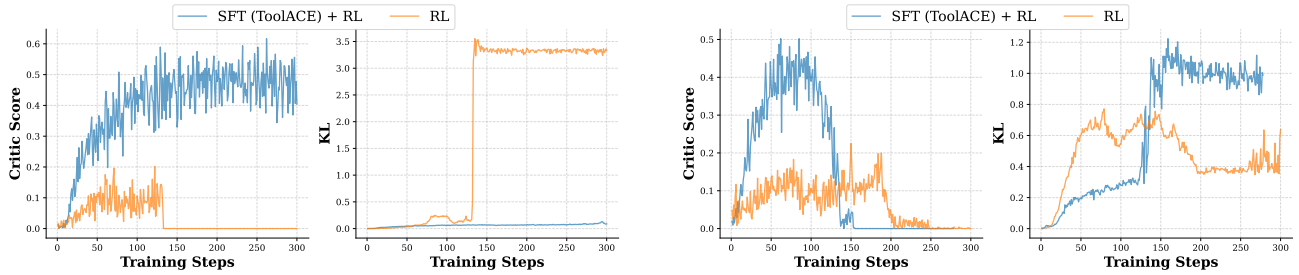


Figure 2 The training dynamics during the training process on the BFCL-V3 dataset. **Left**: Qwen2.5-1.5B-Instruct. **Right:** Qwen3-1.7B.

receive positive feedback. For the reasoning Qwen3-1.7B model, we use its non-thinking version. During RL training, we prepend `<think></think>` before each action to skip the thinking mode. During gradient updates, this thinking prefix is removed, and only the generated content is used for updates. The details please refer to Appendix B.

4.2. Problem of catastrophic collapse in the RL

As shown in Figure 2, direct RL training on Qwen2.5-1.5B-Instruct often leads to **catastrophic collapse**, marked by a sudden drop in reward and a sustained increase in KL divergence. Although Qwen3-1.7B exhibits comparatively milder instability, its performance gains under pure RL remain limited and eventually deteriorate over training. In contrast, applying SFT on an additional tool-use dataset prior to RL stabilizes training dynamics. For Qwen2.5-1.5B-Instruct, reward and KL curves evolve smoothly without severe oscillations. For Qwen3-1.7B, although RL can still experience collapse post-SFT, the model achieves substantial performance improvement before the collapse occurs.

4.3. Analysis of the reasons of collapse

By jointly inspecting model outputs during training, we observe a clear pattern of structural drift. We argue that **the apparent collapse observed during training does not reflect a loss of the model's underlying capability. Rather, it arises from the confounding of markers or specific tokens in current setting of tool calls**, such as `<tool_call>` and `<|im_end|>`, which are misused in combination, leading to structural errors under particular prompt formats. Specifically, for a fixed query, the model's initially correct tool-calling behavior gradually degrades into malformed and ultimately degenerate output formats.

To visualize and analyze this process, we categorize model outputs into four structural states. **Healthy Tool Call** denotes well-formed tool invocations that strictly follow the JSON schema. **Healthy Response** corresponds to coherent natural-language answers that terminate properly and contain no tool-related artifacts. **Text Pollution** represents an early-stage failure mode, in which the model emits incomplete or misplaced tool tags or reasoning tokens, reflecting overfitting to control tokens and a loss of structural coherence. **Collapsed** denotes the most severe degeneration, where the model outputs only a meaningless minimal termination sequence (e.g., `<tool_call><|im_end|>`), effectively bypassing semantic generation.

As shown in Figure 1, Qwen2.5-1.5B-Instruct initially maintains a clear separation between **tool-call** and **text-response** pathways, resulting in predominantly structurally healthy outputs during early training. However, as RL optimization progresses, the model begins injecting the fragmented pattern `<tool_call><|im _end|>` into text responses, signaling an erosion of structural constraints as corresponding examples are shown in Table 3. In later stages, this degradation intensifies, with the model collapsing almost entirely to the minimal termination sequence, which captures the majority of output probability mass. For detailed definitions and additional results, please refer to Appendix A.

This collapse or performance stagnation arises *because the model initially lacks sufficient guidance to consistently sample correct actions*. Consequently, certain special control tokens become disproportionately reinforced in the absence of adequate structural supervision. From the results in Table 2, we observe that in OOD evaluations, performance remains stable for direct GRPO training despite differences in prompt format. In contrast, the SFT-then-RL approach exhibits a noticeable decline under the same conditions. This indicates that the model's underlying tool-use capabilities remain intact, and the observed drop is primarily due to format-specific effects, where the distribution of tool-use tokens in that particular format temporarily masks the model's true abilities.

These observations highlight the format sensitivity of tool-using models. Unlike standard reasoning tasks, tool invocation heavily relies on special control tokens. Without proper supervision or constraints, these tokens can be easily conflated, leading to structural contamination and, ultimately, training collapse, even though the model's underlying reasoning or task-solving ability remains intact.

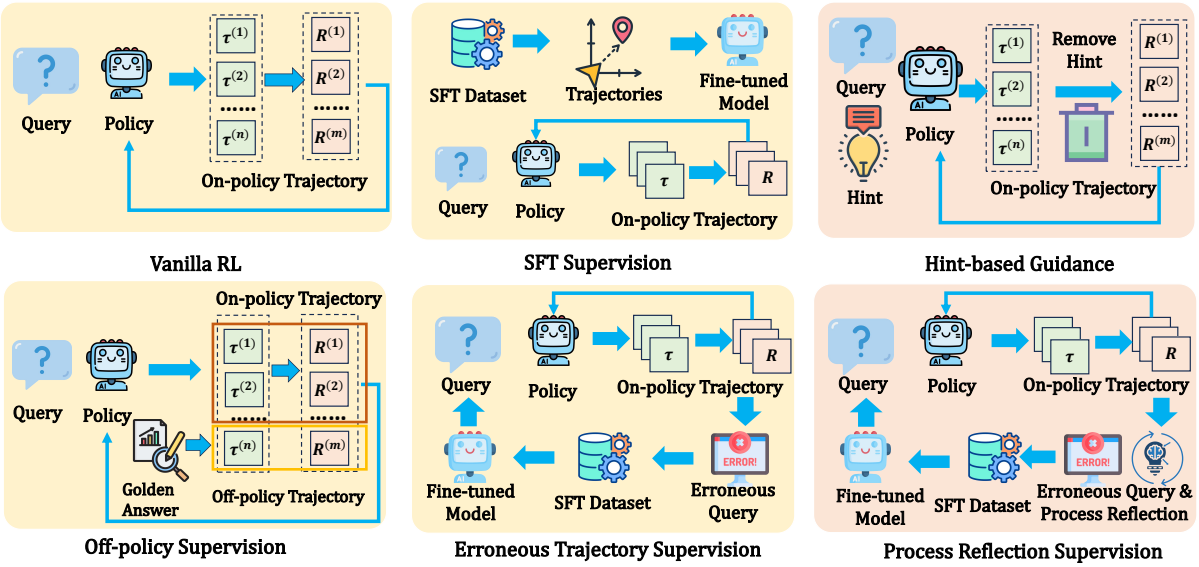


Figure 3 The training frameworks of different methods.

4.4. How to improve the collapse problem

Previous work has demonstrated the effectiveness of SFT for tool-use learning (Liu et al., 2025b; Prabhakar et al., 2025a). Building on this insight, we investigate how different supervisory signals affect agentic training. Specifically, we consider five types of signals: *SFT Supervision (SFT then RL)*, *Off-policy Supervision*, *Hint-based Guidance*, *Erroneous Trajectory Supervision*, and *Process Reflection Supervision*. These signals are integrated under two training paradigms: **synchronous training**, which includes Off-policy Supervision and Hint-based Guidance; and **interleaved training**, which injects Erroneous Trajectory Supervision or Process Reflection Supervision. The structural diagrams of different methods are shown in the Figure 3.

SFT Supervision (SFT then RL) This supervisory signal adopts a staged training approach, where SFT is performed prior to RL. In the SFT stage, the policy π_θ is trained via SFT by imitating expert trajectories τ corresponding to queries q from the dataset D_{SFT}. The training objective maximizes the likelihood of the expert sequence:

$$\mathcal{L}_{\text{SFT}}(\theta) = \frac{1}{|D_{\text{SFT}}|} \sum_{(q,\tau)\in D_{\text{SFT}}} \sum_{t=1}^{|\tau|} -\log \pi_\theta(a_t \mid q, \tau_{<t}) \quad (3)$$

SFT effectively instills instruction-following behavior and basic reasoning skills (e.g., tool invocation formats), but its performance is limited by dataset coverage and lacks exploratory capacity beyond observed trajectories.

Off-policy Supervision (OPS) Inspired by LUFFY (Yan et al., 2025), we introduce off-policy supervision by partially replacing model-generated rollouts with ground-truth trajectories, thereby strengthening the guidance provided by correct behaviors. Concretely, we incorporate off-policy supervision by mixing high-quality SFT trajectories with on-policy rollouts. For trajectory τ_i, the normalized advantage is computed as:

$$\hat{A}_i = \frac{R(\tau_i) - \text{mean}(G_{\text{on}} \cup G_{\text{off}})}{\text{std}(G_{\text{on}} \cup G_{\text{off}})}, \quad (4)$$

where G_{on} and G_{off} are the sets of rewards from on-policy and off-policy trajectories, respectively. Unlike LUFFY that modifies the importance policy ratio for off-policy data, The same policy ratio is applied to all trajectories: $r_{k,t}(\theta) = \pi_\theta(\tau_{k,t} \mid q, \tau_{k,<t}) / \pi_{\theta_{\text{old}}}(\tau_{k,t} \mid q, \tau_{k,<t})$. yielding the off-policy supervision surrogate objective:

$$\mathcal{J}_{\text{OPS}}(\theta) = \frac{1}{Z} \sum_{\tau_i \in \mathcal{D}_{\text{on}} \cup \mathcal{D}_{\text{off}}} \sum_{t=1}^{|\tau_i|} \text{CLIP}\big(r_{k,t}(\theta), \hat{A}_i, \epsilon\big), \quad (5)$$

with $Z = \sum_i |\tau_i|$. This formulation treats on-policy and off-policy trajectories uniformly, allowing high-quality off-policy signals to guide learning while maintaining self-driven exploration.

Hint-based Guidance (HBG) To guide the model towards correct trajectories, we prepend a hint $\mathcal{H}_q$ to the query q during sampling. The hint provides a textual description of how to approach or solve the task, and can be obtained either through manual annotation or generated automatically

by a large language model. At each timestep, the trajectory is generated token by token as:

$$\tau_{i,t} \sim \pi_\theta(\tau_t \mid q, \tau_{i,<t}, \mathcal{H}_q), \tag{6}$$

For policy optimization, the hint $\mathcal{H}_q$ is removed, and the sampled trajectory τ_i is treated as a standard on-policy rollout conditioned only on q. This design allows hints to influence exploration while ensuring that the learned policy does not depend on auxiliary guidance at inference time.

Erroneous Trajectory Supervision (ETS) Inspired by ReLIFT (Ma et al., 2025), which interleaves RL and SFT to overcome the intrinsic limitations of pure RL, we adopt a similar training paradigm in which the SFT stage is explicitly constructed from failure cases encountered during RL. Building on this idea, we explore an error-driven training paradigm that leverages failed samples to further enhance model capabilities.

Starting from an initial policy π_{θ_0}, we first perform RL to collect full interaction trajectories. For each query, if all sampled trajectories fail to complete the task correctly, we identify them as erroneous trajectories. These failures typically arise because correct solutions are not sampled during exploration. We collect such erroneous trajectories and, in a subsequent step, refine the policy using supervised fine-tuning with corresponding ground-truth solutions.

Formally, at iteration k, the correct label training data identified by erroneous trajectories are defined as:

$$\mathcal{E}_k = \bigcup_{q \in Q} \{(q, \tau_{label}) | \forall \tau' \in \mathcal{T}_q, R(\tau') = 0\}, \tag{7}$$

where $\mathcal{T}_q$ denotes the set of trajectories sampled for query q, and $R(\tau)$ is the terminal reward for trajectory. For each erroneous trajectory, we construct supervised training pairs by conditioning on the same state or prefix. The resulting error supervision loss is given by:

$$\mathcal{L}_{\text{err}}(\theta) = -\frac{1}{|\tilde{\mathcal{E}}_k|} \sum_{(q,\tau) \in \tilde{\mathcal{E}}_k} \sum_{t=1}^{|\tau|} -\log \pi_\theta(a_t \mid q, \tau_{<t}), \tag{8}$$

where $\tilde{a}_t$ denotes the correct action derived from ground-truth trajectories.

We interleave this corrective SFT step with standard RL updates, and progressively increase the proportion of RL training steps over iterations, following the setting in ReLIFT. This scheduling strategy gradually enhances the model's exploratory capability while using error-driven supervision to suppress degenerate behaviors. As a result, the policy benefits from both targeted correction of failure modes and sustained improvement in long-horizon tool-use performance.

Process Reflection Supervision (RPS) During multi-turn agentic RL, the model's sampling trajectories contain rich intermediate information beyond terminal success or failure. Prior works (Tan et al., 2025) have explored process rewards to guide optimization. However, scalar process or outcome rewards provide only coarse supervision over content and structure and often fail to capture nuanced reasoning or tool-use errors. We therefore propose *process reflection supervision*, which transforms implicit trajectory information into explicit textual supervision.

Concretely, for trajectories sampled from the current policy π_{θ_k}, we feed the trajectories into an LLM (or an auxiliary analyzer) to generate textual reflections that analyze intermediate decisions, structural mistakes, and reasoning gaps, defined as r_τ^{ref}. We denote the resulting reflection-augmented dataset as:

$$\mathcal{R}_k = \{(q, \tau, r_\tau^{\text{ref}})\}, \tag{9}$$

To further improve the model's adherence to correct tool-use formats, we jointly train with erroneous trajectories collected as described in Section 4.4. The reflection supervision loss is defined as:

$$\begin{aligned} \mathcal{L}_{\text{ref}}(\theta) = &-\frac{1}{|\mathcal{R}_k|} \sum_{(q, r_\tau^{\text{ref}}) \in \mathcal{R}_k} -\log \pi_\theta(r_\tau^{\text{ref}} \mid q) \\ &-\frac{1}{|\tilde{\mathcal{E}}_k|} \sum_{(q,\tau) \in \tilde{\mathcal{E}}_k} \sum_{t=1}^{|\tau|} -\log \pi_\theta(a_t \mid q, \tau_{<t}), \end{aligned} \tag{10}$$

5. Experiments

5.1. Dataset and Models

We train the models on *Qwen2.5-1.5B-Instruct* and *Qwen3-1.7B* and evaluated them on BFCL-V3. To assess generalization, we further tested the models on ACEBench. Detailed training settings are provided in the Appendix B.

5.2. Main Results

Based on the experimental results in Table 1, both Qwen2.5-1.5B-Instruct and Qwen3-1.7B exhibit negligible baseline performance, highlighting the formidable challenges of multi-turn tool-use in the absence of targeted training.

For Qwen2.5-1.5B-Instruct, SFT on BFCL significantly elevates the performance floor, with subsequent GRPO providing further refinement. Conversely, GRPO alone fails to yield meaningful gains. Its inherent stochasticity often triggers catastrophic performance collapse, underscoring that prior structural supervision is an indispensable anchor

Table 1 Experimental results on Qwen series models. Best results within each model group are highlighted in **bold**. Since evaluation is conducted using the final checkpoint, some models may demonstrate strong performance during intermediate training stages but fail to retain these gains at convergence. Detailed training dynamics of individual metrics are provided in the Appendix D.

Model	Method	Base	Δ	Miss F.	Δ	Miss P.	Δ	Long C.	Δ	Average	Δ
Qwen2.5-1.5B	Vanilla	4.0	-	5.0	-	1.0	-	4.0	-	3.50	-
	SFT_{BFCL}	15.0	(+15.0)	4.0	(+14.0)	6.0	(+16.0)	7.0	(+8.0)	16.75	(+13.25)
	GRPO	0.0	(-4.0)	0.0	(-5.0)	0.0	(-1.0)	0.0	(-4.0)	0.0	(-3.5)
	SFT_{BFCL} + RL	21.0	(+17.0)	22.0	(+17.0)	19.0	(+18.0)	7.0	(+3.0)	17.25	(+13.75)
	$SFT_{ToolACE}$ + RL	23.0	(+19.0)	23.0	(+18.0)	13.0	(+12.0)	10.0	(+6.0)	17.25	(+13.75)
	OPS	1.0	(-3.0)	3.0	(-2.0)	1.0	(0)	1.0	(-3.0)	1.50	(-2.0)
	HBG	0.0	(-4.0)	0.0	(-5.0)	0.0	(-1.0)	0.0	(-4.0)	0.0	(-3.5)
	ETS	26.0	(+22.0)	25.0	(+20.0)	16.0	(+15.0)	13.0	(+9.0)	20.0	(+16.5)
	PRS	**31.0**	(+27.0)	**25.0**	(+20.0)	**26.0**	(+25.0)	**21.0**	(+17.0)	**25.75**	(+22.25)
Qwen3-1.7B	Vanilla	14	-	11	-	14	-	11	-	12.5	-
	SFT_{BFCL}	23	(+9)	20	(+9)	23	(+9)	14	(+3)	20	(+7.5)
	$SFT_{ToolACE}$	12	(-2)	10	(-1)	5	(-9)	8	(-3)	8.75	(-3.75)
	GRPO	2	(-13)	0	(-11)	2	(-13)	2	(-10)	1.5	(-11.0)
	SFT_{BFCL} + RL	0	(-14)	0	(-11)	0	(-14)	0	(-11)	0	(-12.5)
	$SFT_{ToolACE}$ + RL	0	(-14)	0	(-11)	0	(-14)	0	(-11)	0	(-12.5)
	OPS	0	(-14)	0	(-11)	0	(-14)	0	(-11)	0	(-12.5)
	HBG	1	(-13)	0	(-11)	1	(-13)	1	(-10)	0.75	(-11.75)
	ETS	**26**	(+12)	**27**	(+14)	**19**	(+5)	**21**	(+10)	**23.25**	(+10.75)
	PRS	23	(+9)	**22**	(+11)	17	(+3)	16	(+5)	19.5	(+7.0)

for agentic RL. Among the examined supervisory signals, Off-Policy Supervision (OPS) and Hint-Based Guidance (HBG) offer limited benefits. Notably, we observed that the KL divergence during synchronous training is substantially higher than that in standard RL procedures in Figure 5. This anomaly likely arises from a mismatch in sampling dynamics: the model's output distribution is forced into rapid shifts while trajectories are not sampled strictly following the current policy, thereby destabilizing the optimization landscape. This observation necessitates a more granular framework to synchronize policy updates and sampling. Furthermore, Process Reflection Supervision (PRS) achieves the most substantial improvements, reaching an average score of 25.75, followed by Erroneous Trajectory Supervision (ETS). Remarkably, although the reasoning-heavy format of PRS diverges from standard tool-calling syntax, it provides critical logical scaffolding that enhances the model's underlying tool-use competence. These findings suggest that interleaved training, which decouples SFT-like structural updates from RL-based exploration, is fundamentally tend to be more stable than synchronous training.

For Qwen3-1.7B, we observe consistent overall trends, but performing SFT prior to RL does not yield strong improvements. As shown in the Figure 15, the model initially achieves higher performance compared to direct RL, yet eventually suffers catastrophic collapse. Based on the analysis in Section C, we attribute this instability to issues in the training sampling format induced by the model's "thinking" mode, which can disrupt structured trajectory learning.

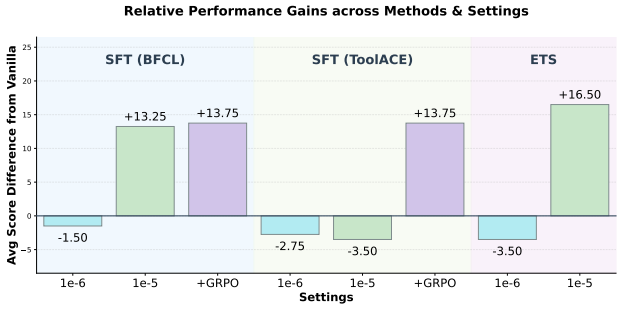


Figure 4 Results on Qwen2.5-1.5B-Instruct under different learning rate. The Y-axis illustrates performance gains and losses relative to the Vanilla baseline (0.0 line), while the X-axis represents various hyperparameter configurations and training stages, with + *GRPO* indicating RL application on models trained at a 1×10^{-5} lr.

5.3. The Analysis of Learning Rate

We evaluate the effect of the learning rate on Qwen2.5-1.5B-Instruct across various training configurations in Figure 4. A conservative learning rate of 1×10^{-6} yields negligible improvements and fails to prevent structural collapse, indicating that overly cautious updates are insufficient to stabilize multi-turn agentic behaviors. Increasing the learning rate to 1×10^{-5} improves performance for SFT (BFCL). In contrast, while SFT (ToolACE) at 1×10^{-5} initially leads to a performance drop due to distribution mismatch, subsequent RL facilitates a substantial recovery and breakthrough. This suggests that fine-tuning on divergent data does not destroy the model's inherent capabilities but rather requires downstream reinforcement learning to reactivate them. Notably, when integrated with Erroneous Trajectory

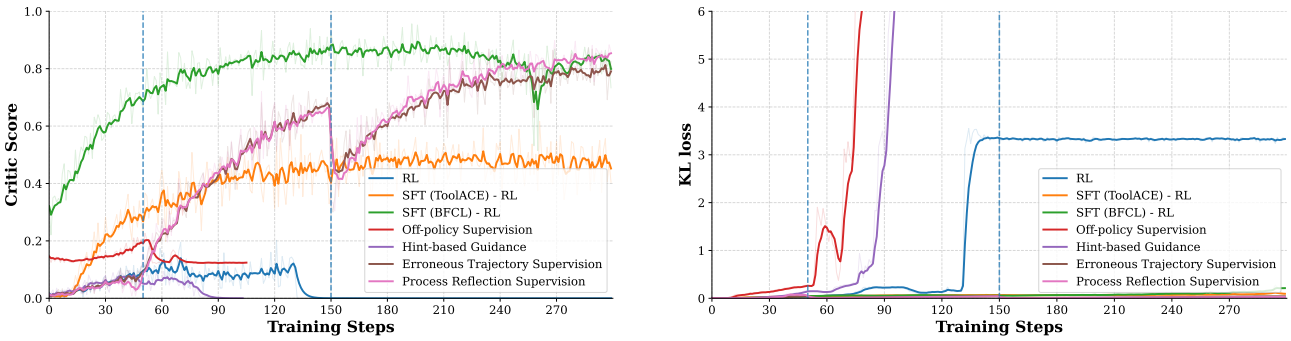


Figure 5 The training dynamics during the training process on the BFCL-V3 dataset of Qwen2.5-1.5B-Instruct.

Supervision (ETS), the larger learning rate (1×10^{-5}) in SFT stage enables significant improvements across all metrics, highlighting that guided, error-driven updates are highly sensitive to the scale of optimization. These observations also suggest that improving stability in multi-turn training may require learning rates larger than those typically used for RL, which could explain instability in synchronous training when using the same learning rate as RL.

5.4. The Analysis of Generalization

We evaluate model generalization under out-of-distribution (OOD) settings using the original ACEBench framework, which features unseen tool types and prompt structures. Many prior studies prioritize scenario generalization without format generalization. We report both generalization axes to analyze the role of supervisory signals.

Defining ID vs. OOD in our setup. We distinguish content OOD from format OOD. Content OOD evaluates models on tool types and scenarios absent from the in-distribution (ID) training data. Format OOD is instantiated by ACEBench's fixed invocation template—calls written as "[ApiName(key1='value1', key2='value2', ...), ...]", whereas ID evaluation (including training) follows Qwen's native format. Therefore, we observe a general performance drop in OOD scenarios, caused by both factors. Experimental results yield several critical insights:

Format- and content-sensitive overfitting during SFT. As Table 2 shows, SFT methods that excel in-distribution suffer sharp degradation under Format and Content OOD in simple single-step settings but not on Content OOD. This supports our claim that standard SFT induces severe format overfitting, tethering the model to training-time syntax.

RL "collapse" as a format artefact. Methods that appear unstable during training (e.g., GRPO-only, OPS, HBG) exhibit more stable behavior under Format and Content OOD tests, while Content OOD in same tool format experiments show decline. This confirms that the observed instability is format-specific (token-level probability shifts) rather than a global loss of reasoning or linguistic competence, as hypothesized in Section 4.3. The Multi Turn results further indicate limited generalization during training.

Process-Level Regularization. While interleaved training can degrade single-turn performance, the integration of Process Reflection Supervision (PRS) effectively mitigates this decline. In addition, these results indicate that isolated SFT stages are more prone to inducing format overfitting: while they can substantially improve in-distribution performance, they often harm generalization unless complemented with process-level supervision. Unlike isolated SFT which prioritizes output-format optimization, PRS encourages the model to internalize the underlying logical scaffolding of tool use. This suggests that incorporating reasoning-heavy, process-oriented data during the SFT phase acts as a crucial logical regularizer, enhancing the model's resilience to distribution shifts in real-world agentic environments.

6. Conclusion

This work studies reinforcement learning for multi-turn tool-use and identifies two key failure modes: training instability and limited performance gains. The analysis reveals that naive RL can over-amplify special control tokens, breaking structured tool invocation and causing collapse or plateaus, highlighting the insufficiency of sparse rewards. We systematically investigate how different supervisory signals can be integrated into agentic RL, comparing synchronous and interleaved training paradigms, and show that interleaved training is markedly more stable and effective. Additionally, we analyze the roles of learning rate, data distribution of SFT data, and generalization, offering a principled framework for stable and effective agentic RL in tool use.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning by enhancing the tool-use capabilities of Large Language Models. There are many potential soci-

Table 2 Generalization results of Qwen2.5-1.5B-Instruct on ACEBench. Left: Format and Content OOD. Right: Content OOD. Best results are in bold, where **Ad.** denotes the Adjust subset, **Sw.** represents the Switch subset, **sf.** indicates single function.

Method	Format and Content OOD					Content OOD				
	Ad.	**Sw.**	**sf.**	**Multi Agent**	**Avg**	**Ad.**	**Sw.**	**sf.**	**Multi Agent**	**Avg**
Vanilla	38.0	20.0	46.0	0.0	26.00	40.0	32.0	52.0	2.5	31.6
SFT_{BFCL}	0.0	0.0	0.0	0.0	0.00	**30.0**	20.0	44.0	2.5	24.1
GRPO	32.0	24.0	**43.0**	0.0	24.75	0.0	0.0	0.0	0.0	0.0
SFT_{BFCL} + RL	0.0	0.0	0.0	0.0	0.00	8.0	8.0	10.0	0.0	6.5
OPS	**36.0**	**24.0**	**43.0**	0.0	**25.75**	20.0	16.0	**50.0**	**6.7**	23.2
HBG	28.0	20.0	42.0	0.0	22.50	0.0	0.0	0.0	0.0	0.0
ETS	10.0	0.0	25.0	0.0	8.75	26.0	22.0	36.0	2.5	21.6
PRS	16.0	8.0	24.0	0.0	12.00	26.0	**30.0**	43.0	4.2	**25.8**

etal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (No.U24A20335), and the independent research project of the Key Laboratory of Cognition and Decision Intelligence for Complex Systems.

References

Chen, S., Lin, S., Gu, X., Shi, Y., Lian, H., Yun, L., Chen, D., Sun, W., Cao, L., and Wang, Q. Swe-exp: Experience-driven software issue resolution. *CoRR*, abs/2507.23361, 2025. doi: 10.48550/ARXIV.2507.23361. URL https://doi.org/10.48550/arXiv.2507.23361.

Chu, T., Zhai, Y., Yang, J., Tong, S., Xie, S., Schuurmans, D., Le, Q. V., Levine, S., and Ma, Y. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Fang, R., Liang, Y., Wang, X., Wu, J., Qiao, S., Xie, P., Huang, F., Chen, H., and Zhang, N. Memp: Exploring agent procedural memory. *CoRR*, abs/2508.06433, 2025. doi: 10.48550/ARXIV.2508.06433. URL https://doi.org/10.48550/arXiv.2508.06433.

Fu, Y., Chen, T., Chai, J., Wang, X., Tu, S., Yin, G., Lin, W., Zhang, Q., Zhu, Y., and Zhao, D. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. *arXiv preprint arXiv:2506.19767*, 2025.

Hao, Y., Cao, P., Jin, Z., Liao, H., Chen, Y., Liu, K., and Zhao, J. CITI: enhancing tool utilizing ability in large language models without sacrificing general performance. In Walsh, T., Shah, J., and Kolter, Z. (eds.), *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pp. 23996–24004. AAAI Press, 2025. doi: 10.1609/AAAI.V39I22.34573. URL https://doi.org/10.1609/aaai.v39i22.34573.

Hu, J. REINFORCE++: A simple and efficient approach for aligning large language models. *CoRR*, abs/2501.03262, 2025. doi: 10.48550/ARXIV.2501.03262. URL https://doi.org/10.48550/arXiv.2501.03262.

Huang, Q., Chan, L., Liu, J., He, W., Jiang, H., Song, M., Chen, J., Yao, C., and Song, J. Boosting MLLM reasoning with text-debiased hint-grpo. *CoRR*, abs/2503.23905, 2025. doi: 10.48550/ARXIV.2503.23905. URL https://doi.org/10.48550/arXiv.2503.23905.

Jiang, D., Lu, Y., Li, Z., Lyu, Z., Nie, P., Wang, H., Su, A., Chen, H., Zou, K., Du, C., Pang, T., and Chen, W. Verltool: Towards holistic agentic reinforcement learning with tool use. *CoRR*, abs/2509.01055, 2025. doi: 10.48550/ARXIV.2509.01055. URL https://doi.org/10.48550/arXiv.2509.01055.

Jin, B., Zeng, H., Yue, Z., Wang, D., Zamani, H., and Han, J. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *CoRR*, abs/2503.09516, 2025. doi: 10.48550/ARXIV.2503.09516. URL https://doi.org/10.48550/arXiv.2503.09516.

Li, M., Zhao, Y., Yu, B., Song, F., Li, H., Yu, H., Li, Z., Huang, F., and Li, Y. Api-bank: A comprehensive benchmark for tool-augmented llms. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 3102–3116. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.187. URL https://doi.org/10.18653/v1/2023.emnlp-main.187.

Li, X., Zou, H., and Liu, P. Torl: Scaling tool-integrated RL. *CoRR*, abs/2503.23383, 2025. doi: 10.48550/ARXIV.2503.23383. URL https://doi.org/10.48550/arXiv.2503.23383.

Liu, M., Farina, G., and Ozdaglar, A. E. UFT: unifying supervised and reinforcement fine-tuning. *CoRR*, abs/2505.16984, 2025a. doi: 10.48550/ARXIV.2505.16984. URL https://doi.org/10.48550/arXiv.2505.16984.

Liu, W., Huang, X., Zeng, X., Hao, X., Yu, S., Li, D., Wang, S., Gan, W., Liu, Z., Yu, Y., Wang, Z., Wang, Y., Ning, W., Hou, Y., Wang, B., Wu, C., Wang, X., Liu, Y., Wang, Y., Tang, D., Tu, D., Shang, L., Jiang, X., Tang, R., Lian, D., Liu, Q., and Chen, E. Toolace: Winning the points of LLM function calling. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025b. URL https://openreview.net/forum?id=8EB8k6DdCU.

Lu, S., Wang, Z., Zhang, H., Wu, Q., Gan, L., Zhuang, C., Gu, J., and Lin, T. Don't just fine-tune the agent, tune the environment. *CoRR*, abs/2510.10197, 2025. doi: 10.48550/ARXIV.2510.10197. URL https://doi.org/10.48550/arXiv.2510.10197.

Ma, L., Liang, H., Qiang, M., Tang, L., Ma, X., Wong, Z. H., Niu, J., Shen, C., He, R., Li, Y., et al. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *arXiv preprint arXiv:2506.07527*, 2025.

Mai, X., Xu, H., Li, Z.-Z., Wang, W., Hu, J., Zhang, Y., Zhang, W., et al. Agent rl scaling law: Agent rl with spontaneous code execution for mathematical problem solving. *arXiv preprint arXiv:2505.07773*, 2025.

Mialon, G., Dessì, R., Lomeli, M., Nalmpantis, C., Pasunuru, R., Raileanu, R., Rozière, B., Schick, T., Dwivedi-Yu, J., Celikyilmaz, A., Grave, E., LeCun, Y., and Scialom, T. Augmented language models: a survey. *Trans. Mach. Learn. Res.*, 2023, 2023. URL https://openreview.net/forum?id=jh7wH2AzKK.

Patil, S. G., Mao, H., Yan, F., Ji, C. C., Suresh, V., Stoica, I., and Gonzalez, J. E. The berkeley function calling leaderboard (BFCL): from tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=2GmDdhBdDk.

Plaat, A., van Duijn, M. J., van Stein, N., Preuss, M., van der Putten, P., and Batenburg, K. J. Agentic large language models, a survey. *CoRR*, abs/2503.23037, 2025. doi: 10.48550/ARXIV.2503.23037. URL https://doi.org/10.48550/arXiv.2503.23037.

Prabhakar, A., Liu, Z., Zhu, M., Zhang, J., Awalgaonkar, T., Wang, S., Liu, Z., Chen, H., Hoang, T., Niebles, J. C., Heinecke, S., Yao, W., Wang, H., Savarese, S., and Xiong, C. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. *CoRR*, abs/2504.03601, 2025a. doi: 10.48550/ARXIV.2504.03601. URL https://doi.org/10.48550/arXiv.2504.03601.

Prabhakar, A., Liu, Z., Zhu, M., Zhang, J., Awalgaonkar, T., Wang, S., Liu, Z., Chen, H., Hoang, T., Niebles, J. C., et al. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. *arXiv preprint arXiv:2504.03601*, 2025b.

Qian, C., Acikgoz, E. C., He, Q., Wang, H., Chen, X., Hakkani-Tür, D., Tur, G., and Ji, H. Toolrl: Reward is all tool learning needs. *CoRR*, abs/2504.13958, 2025. doi: 10.48550/ARXIV.2504.13958. URL https://doi.org/10.48550/arXiv.2504.13958.

Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., Lin, Y., Cong, X., Tang, X., Qian, B., Zhao, S., Hong, L., Tian, R., Xie, R., Zhou, J., Gerstein, M., Li, D., Liu, Z., and Sun, M. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Qu, C., Dai, S., Wei, X., Cai, H., Wang, S., Yin, D., Xu, J., and Wen, J. Tool learning with large language models: a survey. *Frontiers Comput. Sci.*, 19(8):198343, 2025. doi: 10.1007/S11704-024-40678-2. URL https://doi.org/10.1007/s11704-024-40678-2.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y. K., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Srivastava, S. S. and Aggarwal, V. A technical survey of reinforcement learning techniques for large language models. *CoRR*, abs/2507.04136, 2025. doi: 10.48550/ARXIV.2507.04136. URL https://doi.org/10.48550/arXiv.2507.04136.

Tan, W., Qu, X., Tu, M., Ge, M., Liu, A. T., Koehn, P., and Lu, L. Process-supervised reinforcement learning for interactive multimodal tool-use agents.

CoRR, abs/2509.14480, 2025. doi: 10.48550/ARXIV.2509.14480. URL https://doi.org/10.48550/arXiv.2509.14480.

Team, Q. et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3), 2024.

Trivedi, H., Khot, T., Hartmann, M., Manku, R., Dong, V., Li, E., Gupta, S., Sabharwal, A., and Balasubramanian, N. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. In Ku, L., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 16022–16076. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.850. URL https://doi.org/10.18653/v1/2024.acl-long.850.

Wang, Z., Wang, K., Wang, Q., Zhang, P., Li, L., Yang, Z., Jin, X., Yu, K., Nguyen, M. N., Liu, L., et al. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025.

Wu, J., Zhu, J., Liu, Y., Xu, M., and Jin, Y. Agentic reasoning: A streamlined framework for enhancing llm reasoning with agentic tools. *arXiv preprint arXiv:2502.04644*, 2025.

Xie, J., Zhang, K., Chen, J., Zhu, T., Lou, R., Tian, Y., Xiao, Y., and Su, Y. Travelplanner: A benchmark for real-world planning with language agents. *arXiv preprint arXiv:2402.01622*, 2024.

Xue, Z., Zheng, L., Liu, Q., Li, Y., Zheng, X., Ma, Z., and An, B. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *CoRR*, abs/2509.02479, 2025. doi: 10.48550/ARXIV.2509.02479. URL https://doi.org/10.48550/arXiv.2509.02479.

Yan, J., Li, Y., Hu, Z., Wang, Z., Cui, G., Qu, X., Cheng, Y., and Zhang, Y. Learning to reason under off-policy guidance. *CoRR*, abs/2504.14945, 2025. doi: 10.48550/ARXIV.2504.14945. URL https://doi.org/10.48550/arXiv.2504.14945.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Yao, S., Shinn, N., Razavi, P., and Narasimhan, K. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *CoRR*, abs/2406.12045, 2024. doi: 10.48550/ARXIV.2406.12045. URL https://doi.org/10.48550/arXiv.2406.12045.

Zhan, R., Li, Y., Wang, Z., Qu, X., Liu, D., Shao, J., Wong, D. F., and Cheng, Y. Exgrpo: Learning to reason from experience. *CoRR*, abs/2510.02245, 2025. doi: 10.48550/ARXIV.2510.02245. URL https://doi.org/10.48550/arXiv.2510.02245.

Zhang, F., Tan, Z., Ma, X., Dong, Z., Leng, X., Zhao, J., Sun, X., and Yang, Y. Adhint: Adaptive hints with difficulty priors for reinforcement learning. *arXiv preprint arXiv:2512.13095*, 2025a.

Zhang, G., Geng, H., Yu, X., Yin, Z., Zhang, Z., Tan, Z., Zhou, H., Li, Z., Xue, X., Li, Y., et al. The landscape of agentic reinforcement learning for llms: A survey. *arXiv preprint arXiv:2509.02547*, 2025b.

Zhang, K., Lv, A., Li, J., Wang, Y., Wang, F., Hu, H., and Yan, R. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *arXiv preprint arXiv:2507.02841*, 2025c.

Zhang, S., Dong, Y., Zhang, J., Kautz, J., Catanzaro, B., Tao, A., Wu, Q., Yu, Z., and Liu, G. Nemotron-research-tool-n1: Exploring tool-using language models with reinforced reasoning. *CoRR*, abs/2505.00024, 2025d. doi: 10.48550/ARXIV.2505.00024. URL https://doi.org/10.48550/arXiv.2505.00024.

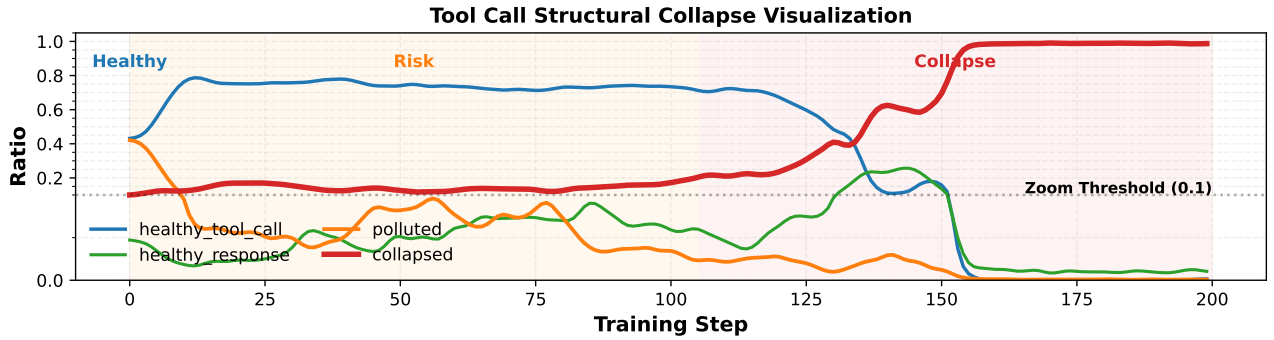


Figure 6 The changes of token appearing frequency in the SFT then RL training process of model Qwen3-1.7B.

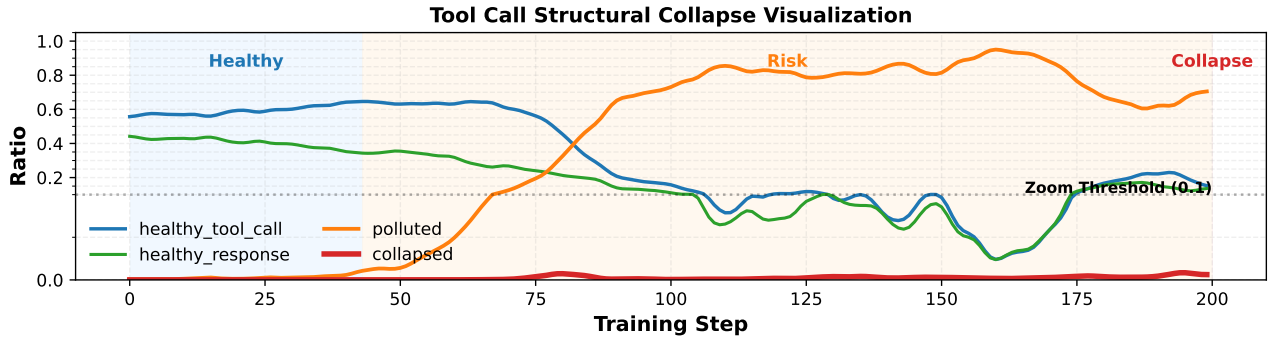


Figure 7 The changes of token appearing frequency in the RL training process of model Qwen3-1.7B.

A. Analysis

Table 3 Examples of different output types observed during multi-turn tool-use training.

Type	Example
Healthy Tool Call	<tool_call>{``name'': ``search'', ``arguments'': {...}}</tool_call><|im_end|>
Healthy Response	The answer can be derived by first querying the database...<|im_end|>
Polluted Output	The result is ...<tool_call><|im_end|>
Collapsed Output	<tool_call><|im_end|>

The detailed classification of model output structures:

- Healthy Tool Call: Model output contains a valid tool invocation: both opening <tool_call> and closing </tool_call> tags are present and correctly matched. The internal action is valid, the structure conforms to the expected XML/JSON schema, and no stray <|im_end|> tokens or partial tags appear within the invocation.

- Healthy Response: Model output does **not** include any tool-call tags (<tool_call>, </tool_call>) but ends with a valid end-of-turn token <|im_end|>. The content is a coherent natural language response without extraneous or misused tool tags.

- Text Polluted: Outputs with partially misused tool-call tags, such as missing closing tags, adding <think>, embedded <|im_end|> inside a tool invocation, or extra tags appended to normal text. Indicates structural corruption where the model mixes dialogue text with tool invocation tags incorrectly, potentially causing protocol violations or parsing errors.

- Collapsed: Outputs consisting solely of the end-of-turn token <|im_end|>, or cases where all tool invocations collapse to just <|im_end|> without any semantic content. Reflects severe RL failure, where the model short-circuits the interaction to maximize reward or avoid penalties, producing no meaningful action or response.

The more analysis results are in Figure 6 and Figure 7.

B. Training Details

Table 4 Supervised Fine-Tuning (SFT) configuration.

Parameter	**SFT**
Epochs	3
Train batch size	128
Learning rate	1e-5
Max sequence length	8192

Table 5 Reinforcement Learning (RL) configuration.

Parameter	**RL**
Total training steps	300
Batch size	60
PPO mini-batch size	20
Learning rate	1e-6
Max turns per trajectory	30
KL loss coefficient	0.0001

The SFT and RL parameter configurations are summarized in the Table 4 and Table 5. The experimental setups for the different supervisory interventions are as follows: for off-policy supervision, 7 trajectories are sampled from the model and 1 trajectory uses the ground-truth label; for hint-based guidance, 6 trajectories are generated normally and 2 are conditioned on hints. For interleaved training, the RL rounds for Qwen2.5-1.5B-Instruct are 50, 100, and 150, while for Qwen3-1.7B, they are 50, 50, and 100.

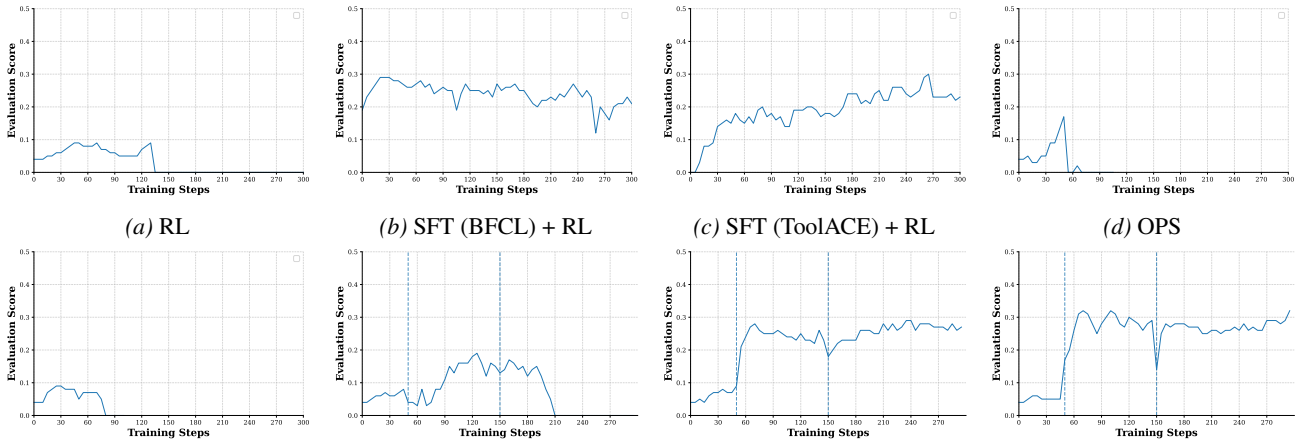
C. The Prompts and Examples of Training

We believe that the performance collapse observed in Qwen3 after the SFT-then-RL pipeline is primarily caused by inconsistencies in prompt formatting. Specifically, Qwen3 is designed to operate in a thinking mode, where the model is expected to generate an explicit reasoning segment. When reasoning output is disabled, the prompt must explicitly include a placeholder such as <think></think>. However, during the SFT stage, the training data contains only direct tool-call outputs without this reasoning wrapper. If, during the subsequent RL stage, the prompt is modified to include the thinking-related tokens while the supervised data does not reflect this structure, a mismatch is introduced between the SFT distribution and the RL sampling format. This inconsistency can destabilize training, leading to structural drift and eventual performance collapse. The results in Figure 8 demonstrate this point.

D. Detailed Evaluation

Since evaluation is conducted using the final checkpoint, some models may demonstrate strong performance during intermediate training stages but fail to retain these gains at convergence. We report the evaluation results in the training process as follows. The results of Qwen2.5 is shown in Figure 9, Figure 10 and Figure 11. The results of Qwen3 is shown in Figure 12, Figure 13 and Figure 14.

Figure 8 The training reward with or without thinking token



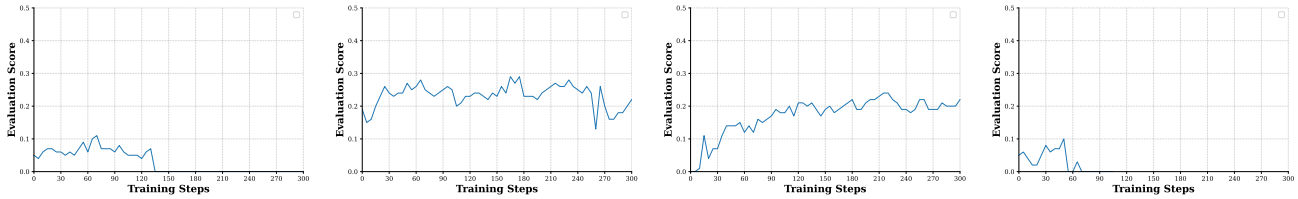
(a) RL *(b)* SFT (BFCL) + RL *(c)* SFT (ToolACE) + RL *(d)* OPS

(e) HBG *(f)* ETS (SFT 1e−6) *(g)* ETS (SFT 1e−5) *(h)* PRS

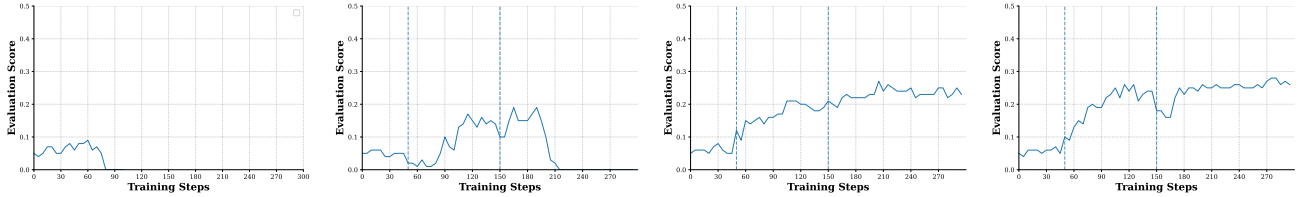
Figure 9 Evaluation results of scenario *Base* under different supervisory signals of Qwen2.5-1.5B-Instruct.

E. Training Dynamic

The Training dynamics of Qwen2.5 are shown in Figure 5 and Qwen3 are shown in Figure 15.

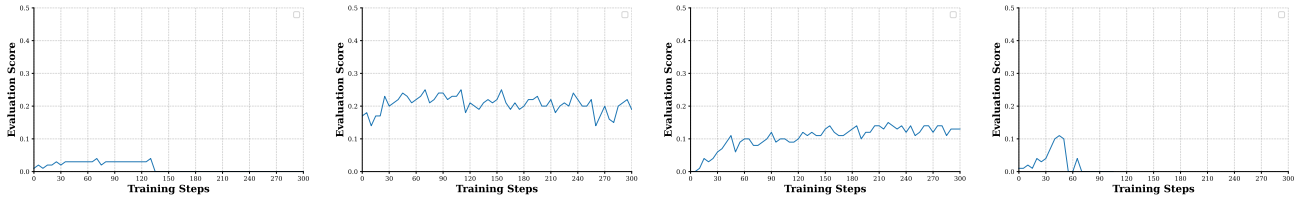


(a) RL *(b)* SFT (BFCL) + RL *(c)* SFT (ToolACE) + RL *(d)* OPS

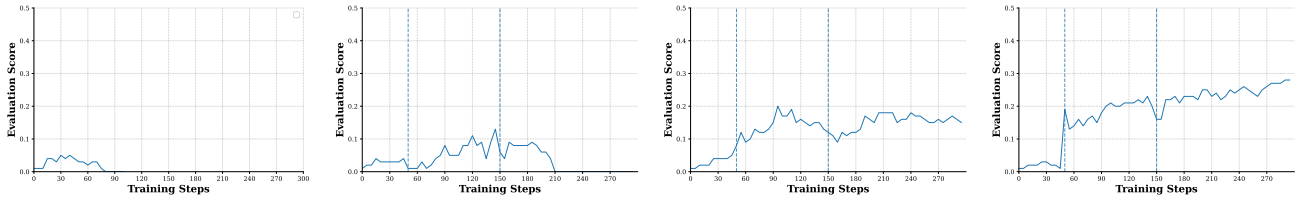


(e) HBG *(f)* ETS (SFT 1e−6) *(g)* ETS (SFT 1e−5) *(h)* PRS

Figure 10 Evaluation results of scenario *Miss Func* under different supervisory signals of Qwen2.5-1.5B-Instruct.

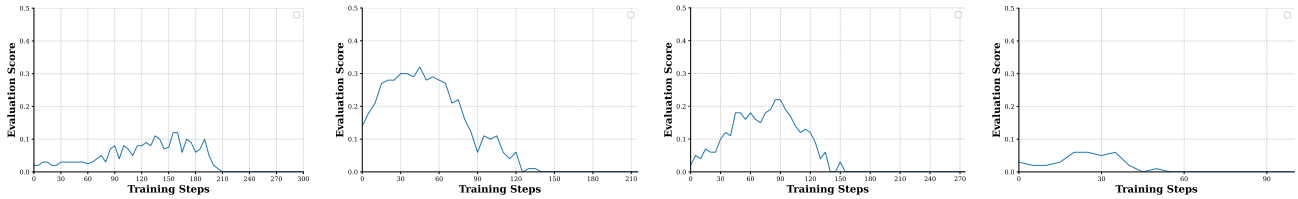


(a) RL *(b)* SFT (BFCL) + RL *(c)* SFT (ToolACE) + RL *(d)* OPS

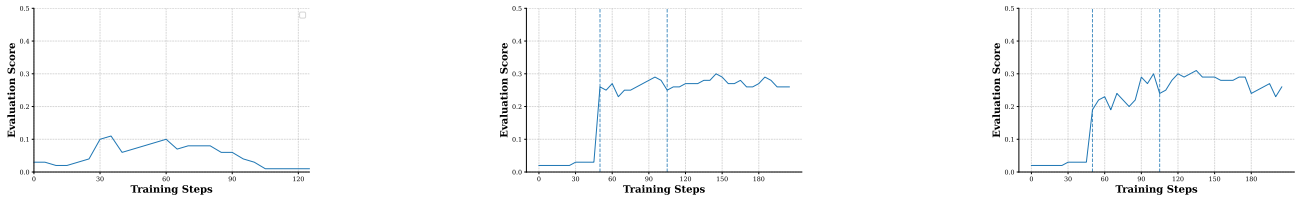


(e) HBG *(f)* ETS (SFT 1e−6) *(g)* ETS (SFT 1e−5) *(h)* PRS

Figure 11 Evaluation results of scenario *Miss Param* under different supervisory signals of Qwen2.5-1.5B-Instruct.

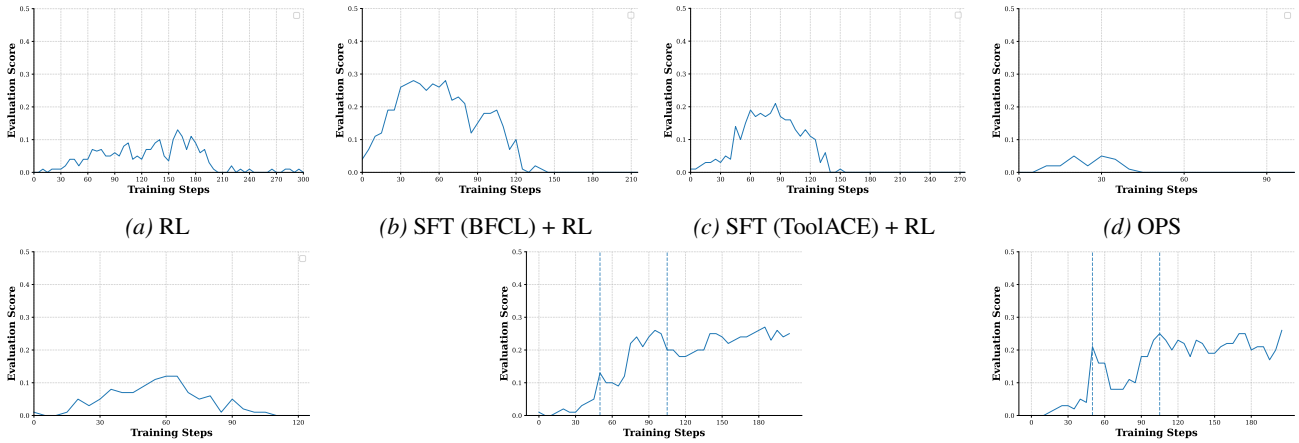


(a) RL *(b)* SFT (BFCL) + RL *(c)* SFT (ToolACE) + RL *(d)* OPS



(e) HBG *(f)* ETS *(g)* PRS

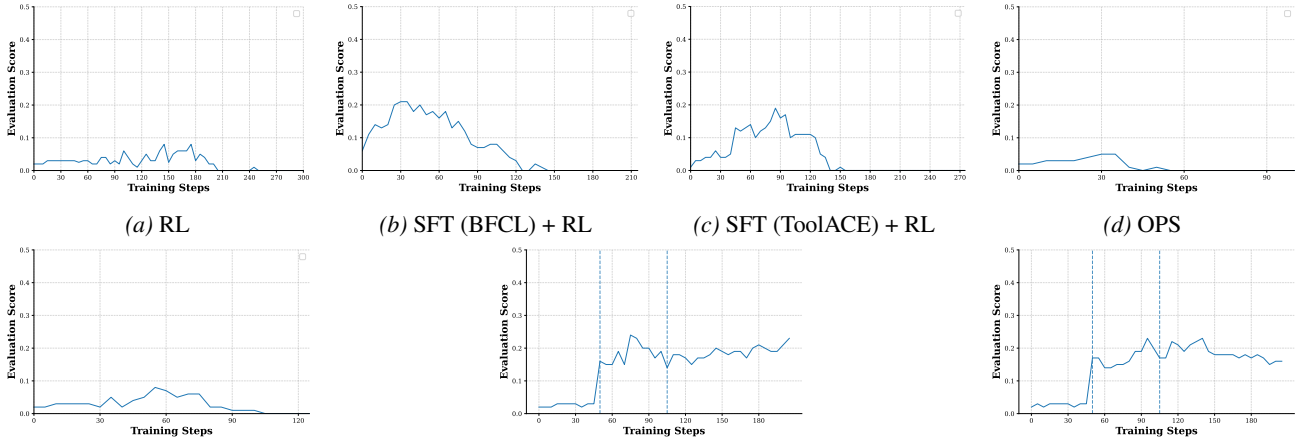
Figure 12 Evaluation results of scenario *Base* under different supervisory signals of Qwen3-1.7B.



(a) RL *(b)* SFT (BFCL) + RL *(c)* SFT (ToolACE) + RL *(d)* OPS

(e) HBG *(f)* ETS *(g)* PRS

Figure 13 Evaluation results of scenario *Miss Func* under different supervisory signals of Qwen3-1.7B.



(a) RL *(b)* SFT (BFCL) + RL *(c)* SFT (ToolACE) + RL *(d)* OPS

(e) HBG *(f)* ETS *(g)* PRS

Figure 14 Evaluation results of scenario *Miss Param* under different supervisory signals of Qwen3-1.7B.

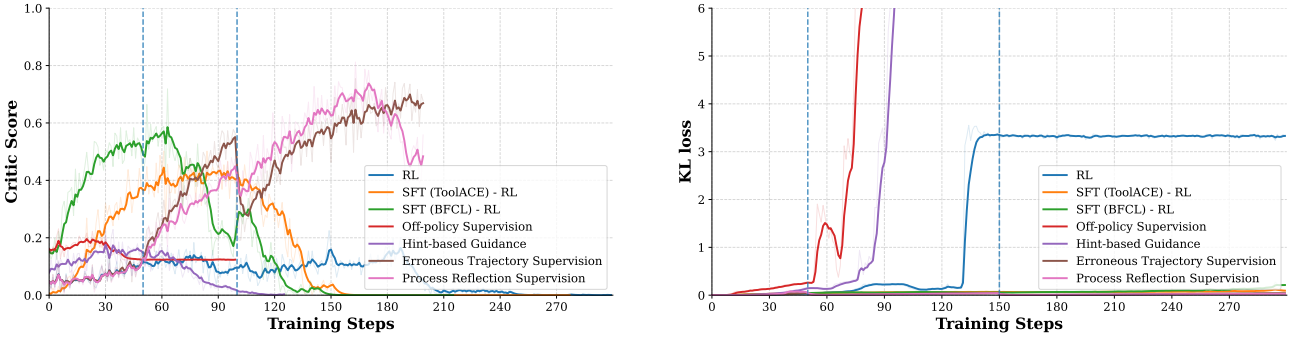


Figure 15 The training dynamics during the training process on the BFCL-V3 dataset of Qwen3-1.7B.

F. Prompt and Example

The prompt of RPS is in Prompt 1 and the example is in Example 2.

```
You are an expert in intelligent agent tool-use analysis.

Below is the erroneous interaction between a model and tools (including the
ground-truth tool-use sequence for reference). Your task is to provide a
**comprehensive, layered, and actionable analysis**, and **generalize to similar
scenarios** to create augmentation data.
**Note:** You should write in natural language (not structured JSON), but every
correct tool call must strictly follow the raw <tool_call> ... </tool_call> format.

Inputs to be filled by caller:
- Available Tools: {available_tools}
- Model Interaction Log (Error Turn): {formatted_summary}

Your required outputs:

1. **Core Error Analysis (must be first)**
  - Briefly state the root cause of the error (1-2 lines)
  - Provide 2-4 pieces of evidence from the interaction log to support your
  conclusion
  - Provide one immediate and actionable fix recommendation

2. **Generalization: Generate 3-5 similar scenarios**
  For each scenario, describe in natural paragraphs:
  - One-sentence scenario description (what & why it occurs)
  - The user request and necessary context
  - A typical model mistake that might happen
  - **The correct tool call sequence**
    (each step must use the raw `<tool_call>\n{{...}}\n</tool_call>` block,
    containing exactly one JSON object with only `name` and `arguments`)
  - What reasoning procedures can be borrowed from the current failed case
  - What aspects differ and require different handling
  - A brief validation strategy for confirming the fix (auto or manual)

  The scenarios must cover different error categories:
  - Missing parameters
  - Wrong tool selection
  - Output mismatch
  - Timeout/failure
  - Ambiguous request or multi-intent

3. **Provide one scenario that is similar but requires a different solution**
  - Explain the similarity (shared reasoning or decision features)
  - Explain clearly why different tool(s) are required
  - Provide correct tool calls in the same `<tool_call>` format
  - Provide rationale

4. **Output style and quality requirements**
  - Natural language, clear paragraph structure
  - No JSON schemas, no markdown code fences, no meta-explanations
  - Tool calls must be valid JSON and executable
  - Provide 1-2 acceptance criteria per scenario for filtering low-quality
  synthetic data

5. **Example (format reference only, do not repeat unless needed)**:
<tool_call>
{{"name": "search_database", "arguments": {{"query": "latest model evaluation",
"top_k": 5}}}}
</tool_call>
```

```
---

Please begin by analyzing the root cause of the provided error turn, then
generalize to 35 similar scenarios (including at least 1 scenario that requires
different handling).
```

Prompt 1 The RPS prompt

```
Turn 1
1) User intent understanding, correct behavior, and success points
- Intent: Price-check then book a business-class SFLA flight for Robert Trenton on
2024-11-25 using travel card card_3487 (access code 1293), and obtain an invoice to
verify charges.
- Correct behavior: (a) get flight cost to confirm price, (b) book the flight with
provided card and access token, (c) retrieve the invoice for verification.
- Success points: All three tools were called in the correct order. Responses show
cost $400, booking_id 3426812 and transaction_id 45451592, and retrieve_invoice
returned a consistent invoice with cost and booking details. This aligns with the
users intent and ground truth.

2) Mistake classification and root cause analysis
- Mistakes: None meaningful. Actions, arguments, and sequence match the
ground-truth sequence exactly.
- Root cause: N/A.

3) Improvement plan (corrected reasoning and ideal tool call sequence)
- Reasoning: Confirm price before booking to avoid unexpected charges; then book
using provided card and access token; finally retrieve the invoice to verify
charges and record transaction details.
- Ideal tool call sequence (matches ground truth and log):
<tool_call>
{"name": "get_flight_cost", "arguments": {"travel_from": "SFO", "travel_to": "LAX",
"travel_date": "2024-11-25", "travel_class": "business"}}
</tool_call>
<tool_call>
{"name": "book_flight", "arguments": {"access_token": "1293", "card_id":
"card_3487", "travel_date": "2024-11-25", "travel_from": "SFO", "travel_to": "LAX",
"travel_class": "business"}}
</tool_call>
<tool_call>
{"name": "retrieve_invoice", "arguments": {"access_token": "1293", "booking_id":
"3426812"}}
</tool_call>

- Alignment: This is identical to the ground truth sequence and satisfies the users
requirement to book and verify the invoice.

Turn 2
1) User intent understanding, correct behavior, and success points
- Intent: Post a tweet on Robert Trentons travel account: "Loved my flight
journey!" with hashtag #TravelDiaries, then retweet it from his travel account to
maximize visibility.
- Correct behavior: Before calling post_tweet/retweet, ensure the Twitter account
is authenticated (credentials or an authenticated session) and that the correct
account is targeted for posting/retweeting. If credentials or an authenticated
account identifier are missing, request them rather than calling tools.

2) Mistake classification (fine-grained) and root cause analysis
- Mistake classification:
```

```
  - Missing Parameter / Precondition Check Failure: The model attempted post_tweet
  without confirming authentication or asking for credentials.
  - Incorrect remediation flow: After failed authentication attempt using wrong
  credentials, the model tried retweeting (still unauthenticated), producing further
  errors.
- Root cause:
  - The agent did not treat authentication as a required precondition; it attempted
  actions without validating credentials or asking for clarifying information
  (username, which account to use).
  - It also attempted to remediate by guessing credentials ("user"/"pass") rather
  than requesting valid credentials from the user.

3) Improvement plan (corrected reasoning and ideal tool call sequence)
- Corrected reasoning: When a user requests an action that requires authentication
but does not provide credentials or indicate an authenticated session, the agent
must pause and request the missing authentication information (or ask the user to
confirm which authenticated account to use). No tool calls should be made until
valid credentials or an authenticated session are provided.
- Action to take now (at this turn): Ask for the necessary authentication details
and account identifier. Do not call any tools until the user supplies credentials
or instructs a currently-authenticated account to be used.
- No tool calls should be made at this turn. If the user supplies credentials, the
correct sequence (ground truth) would be:
<tool_call>
{"name": "authenticate_twitter", "arguments": {"username": "john", "password":
"john1234"}}
</tool_call>
<tool_call>
{"name": "post_tweet", "arguments": {"content": "Loved my flight journey!"}}
</tool_call>
<tool_call>
{"name": "retweet", "arguments": {}}
</tool_call>
- How this aligns with ground truth and logic: Ground truth specifies that because
of missing parameters, no tool calls should be made; instead, the agent should
request credentials. The three-step sequence above is the correct follow-up once
credentials are provided (authenticate  post  retweet).

Turn 3
1) User intent understanding, correct behavior, and success points
- Intent: The user supplied Twitter credentials; authenticate with those
credentials, post the tweet, and retweet it.
- Correct behavior: Authenticate using username/password, post the requested tweet,
then retweet to amplify it.
- Success points: The agent authenticated successfully (authentication_status:
true), posted the tweet ("Loved my flight journey!"), and retweeted successfully.
The returned post id (10) and retweet success indicate the actions worked and
satisfy the users request.

2) Mistake classification and root cause analysis
- Mistakes: Minor inconsistency only:
  - The ground-truth post_tweet call used just content (no explicit tags). In the
  log the tweet included tags in earlier attempts; the final posted tweet in the log
  did include tags in earlier step but ground truth shows no tags in the tool call.
  This is a minor divergence in arguments, not a functional erroruser asked for
  hashtag #TravelDiaries, so including the tag is acceptable.
  - The ground truth retweet call had empty arguments; the agent used tweet_id in
  the log. Either form is acceptable depending on API design; using tweet_id is
  explicit and clear.
- Root cause: Small differences in how the API was used (presence/absence of tags
field, different retweet parameter shapes). These do not change outcome.

3) Improvement plan (corrected reasoning and ideal tool call sequence)
```

```
- Corrected reasoning: Authenticate with supplied credentials; once authenticated,
post the tweet exactly as requested (include hashtag #TravelDiaries if the user
specified it), capture the returned tweet id, then retweet that tweet to increase
visibility. Confirm success and return post and retweet statuses to user.
- Ideal tool call sequence (matches ground truth intent; any of the two retweet
argument styles is acceptable depending on APIhere we show the ground-truth calls
and the explicit tweet-id approach):
<tool_call>
{"name": "authenticate_twitter", "arguments": {"username": "john", "password":
"john1234"}}
</tool_call>
<tool_call>
{"name": "post_tweet", "arguments": {"content": "Loved my flight journey!"}}
</tool_call>
<tool_call>
{"name": "retweet", "arguments": {}}
</tool_call>
- If using the explicit tweet id returned by post_tweet (recommended for clarity):
<tool_call>
{"name": "retweet", "arguments": {"tweet_id": 10}}
</tool_call>

- Alignment: This aligns with the ground-truth successful sequence: authenticate
post_tweet  retweet. The explicit tweet_id variant ensures the retweet targets the
correct post.

Summary Recommendations (cross-turn)
- Always validate preconditions: check for required credentials or tokens before
calling authentication-protected tools. If missing, request them.
- Follow a predictable pattern for transactional flows: estimate cost  confirm
booking  retrieve invoice.
- When a user supplies credentials, authenticate then perform actions; do not guess
credentials.
- When API shapes differ (e.g., retweet accepting {} vs tweet_id), prefer explicit
parameters (tweet_id) if available to avoid ambiguity.
```

Prompt 2 The example of RPS